COHERENT MEMORY MAPPING TABLES FOR HOST I/O BRIDGE

FIELD OF THE INVENTION

[0001] The invention relates to the field of input-output (I/O) address space mapping hardware for computer systems. In particular, the invention relates to I/O address translation devices capable of automatically maintaining coherency in computer systems having multiple DMA-capable peripherals and potentially multiple processors.

BACKGROUND OF THE INVENTION

[0002] Most modern computer systems use virtual memory addressing. These systems compute effective addresses of operands in a virtual memory space. An address translation system then maps portions of the virtual memory space to actual data storage. The actual data storage typically includes one or more swap files or partitions as well as RAM memory. The advantages of these systems are well known in the art.

[0003] Virtual memory systems typically allocate memory in discrete blocks known as pages. When a program references memory at a particular virtual address, that address is translated to determine whether the associated real memory page is in memory, or located in a swap file. If the address corresponds to data in a swap file, real memory is allocated and loaded from the swap file. When the address corresponds to data in real memory (possibly after loading memory from a swap file), the address is translated to a physical memory address in the associated page of real, physical, memory. The reference is then allowed to take place.

[0004] A subset of these modern computer systems also use virtual addressing for Input-Output (I/O) devices. In these systems, a virtual I/O address space is created for Direct Memory Access (DMA) operations. This virtual I/O address space is typically used for I/O buffers, including disk data buffers. These buffers may each comprise contiguous locations in virtual I/O address space, but may map to discontiguous locations in physical memory. Typically, DMA-capable peripherals such as disk interface adapters transfer blocks of data between themselves and memory, generating virtual I/O addresses to address the memory.

[0005] A subset of computer systems supporting virtual I/O addressing, including those supporting the InfiniBand I/O interconnect system (InfiniBand documents are available at http://www.infinibandta.org), also support Remote Direct Memory Access (RDMA)

operations. In these systems, DMA operations may be initiated by peripherals that are remotely located, they may, but need not, be physically located in the same cabinet as the processor and memory. The InfiniBand architecture requires support for RDMA read, write, and atomic operations to buffers in a virtual I/O address space. The term DMA-capable herein includes RDMA-capable devices.

[0006] Such systems have hardware and software for address translation between DMA-capable I/O devices and physical memory to support virtual I/O addresses. I/O address translation of this type is a requirement of systems supporting the InfiniBand I/O architecture.

[0007] Systems supporting address translation between DMA-capable I/O devices and system memory allow for rapid and convenient dynamic allocation of memory buffers and I/O cache space. In these systems, it is possible to configure a DMA-capable I/O device, such as a disk drive, to transfer a block of data. Address translation software and hardware then allows the transfer to take place even if the block of data straddles multiple discontinuous pages of real memory.

[0008] An I/O adapter having address translation capability is described in U.S. Patent 5,784,708, hereinafter the '708 patent, the disclosure of which is hereby incorporated by reference. The input-output adapter of the '708 patent incorporates an I/O Translation Lookaside Buffer (TLB). A TLB is a hardware mapping device that stores part, or all, of a page table used for address translation. Typically, a TLB stores one or more entries of the page table, thereby permitting rapid access to those page table entries, while the full page table is stored in memory. When an I/O device specifies target addresses in memory, those addresses are checked against the page table entries stored in the TLB; if a corresponding page table entry is found those addresses are translated according to the page table entry and the requested transfer occurs. If no corresponding page table entry is found, the TLB is updated from memory with the corresponding page table entry before the transfer occurs.

[0009] Modern computer systems often have multiple DMA-capable I/O devices, and may have multiple processors. These systems may have multiple I/O TLBs. It is known that page tables of such systems may change as the system operates and memory pages are allocated, moved, pushed out into swap file, fetched from swap files, and deallocated. It is also known that page tables of such systems may change as a video bitmaps are replaced with updated, alternate, bitmaps.

[0010] A TLB is considered coherent with a page table in memory if each map entry in the TLB refers to the same place in real memory or swap file as corresponding entries in

the page table data in memory. Multiple TLBs are considered coherent with each other if each is coherent with the same page table in memory.

[0011] It is desirable that each I/O TLB of the system be kept coherent with its associated page table data in memory, regardless of whether each I/O TLB uses a common or a unique page table in memory.

[0012] It is known that the coherency of I/O TLBs with their associated page tables in memory can be maintained through software, and this is common practice in the art. Software maintenance of coherency can, however, consume considerable processor cycles, during which time the processor is unable to perform other useful work. It is therefore desirable to improve, indeed automate, the maintenance of TLB coherency.

[0013] Coherency is also required in cache memory of computer systems, especially in those having multiple processors. There are several known protocols for maintaining cache coherency in multiple processor cache-equipped systems. One such protocol is known as "snooping." Snooping involves having each processor cache of the system monitor a bus for cacheline accesses by other caches of the system. When a cache sees an access to a cacheline it holds, it returns or invalidates its copy of the cacheline if necessary.

[0014] Another known protocol for maintaining cache coherency is "directory-based". In these systems, a central controller maintains a cache directory, a directory of cachelines "owned" by each cache in the system; a cacheline being "owned" by a cache if that cache has a copy of associated data. All accesses of data in the system are cleared against the cache directory to ensure that "dirty" data is written to memory and to ensure that prior owning caches are invalidated as needed. This may be done by messaging each prior owning cache when ownership of a cacheline changes.

[0015] Another form of directory-based cache coherency protocol is one in which a memory system stores a cache state for each cacheline. This cache state includes information about caches owning the associated cacheline, and is maintained by a system controller. With this system, the system controller messages each prior owning cache when ownership of a cacheline changes.

[0016] While snoop-based and directory-based coherency maintenance mechanisms have historically been used to maintain cache coherency, they are not typically used to maintain coherency of address mapping hardware such as TLB's.

[0017] A bridge is a device that couples through a first port to a first computer interconnect and through a second port to a second computer interconnect. The first and

second interconnect may, but need not, be of different types, and are often, but need not be, busses. Bridges transfer data between the first and second ports. For example, a typical computer system has a host bridge that couples a processor's local bus to a PCI or other I/O local bus, and perhaps a second bridge that couples the PCI bus to an ISA bus. A bridge may also couple the interconnect of a host computer to the internal interconnect of a peripheral device. For example, a peripheral device card may incorporate a bridge that couples a PCI bus and connector to an internal PCI bus, the internal PCI bus couples a processor and other devices local to the peripheral device.

[0018] In modern computer systems, a memory system may attach directly to a processor's local bus, or the memory controller and host bridge may be integrated as a single component.

[0019] Bus bridges typically include at least a first and a second interconnect interface, with state machines for transmitting and receiving data on either interconnect interface. Bus bridges typically have address window hardware, such that they respond to a particular set of addresses on the first bus interface, and to a particular set of addresses on the second bus interface. They also have buffers capable of receiving data on the first bus interface, and transmitting it on the second bus interface, and of receiving data on the second bus interface, and transmitting it on the first bus interface.

[0020] I/O address translation logic can be located in a bus bridge. Systems exist wherein I/O address translation logic is located in a host bridge, where the I/O translation logic serves to translate virtual I/O addresses received over the I/O bus into physical addresses in the memory system.

SUMMARY OF THE INVENTION

[0021] A computer system is provided with I/O address mapping hardware. In a particular embodiment, this I/O address mapping hardware is located in a host bridge. In a particular embodiment, this address mapping hardware incorporates an I/O TLB.

[0022] The address mapping hardware is provided with apparatus for maintaining coherency between its mapping entries, and a page table in memory. In a particular embodiment, this apparatus for maintaining currency includes apparatus for snooping.

[0023] In another embodiment, a host channel adapter has I/O address mapping hardware that incorporates an I/O TLB. In this embodiment, the I/O address mapping hardware cooperates with a directory-based coherency protocol, such that mappings stored in

the I/O TLB are invalidated when their corresponding page table entries in memory are modified by some other cache in the system.

BRIEF DESCRIPTION OF THE DRAWINGS

[0024] Figure 1 is a block diagram of a system having two processors, memory, and I/O bus coupled by a host bridge, wherein I/O address translation is performed at the host bridge, and the I/O address translation logic is equipped to snoop the processor-memory (system) bus;

[0025] Figure 2, a block diagram illustrating part of a bridge, detailing a portion of a Translation Lookaside Buffer (TLB) having snoop capability; and

[0026] Figure 3, a block diagram of a system having two intelligent DMA-capable peripheral devices connected through separate host channel adapters (HCAs) coupled to a system controller, wherein address translation logic is located in each HCA, and coherency is maintained using a directory-based protocol.

DETAILED DESCRIPTION OF THE EMBODIMENTS

[0027] In an computer system (Figure 1), there are one or more processors 300, 301. Each processor 300, 301 is coupled to memory 302 over a system bus 306. System bus 306 is coupled through a host bridge 310 to an I/O bus 314. I/O devices, which may include intelligent peripherals 316, 318, are coupled to the I/O bus 314. I/O address translation hardware 320 includes a TLB, and having coherency maintenance logic as described with reference to Figure 2, is located in host bridge 310. I/O address translation hardware 320 maps I/O virtual addresses of memory references originating at peripherals 316, 318 into physical addresses in memory 302. A page table 324 exists in memory 302. The address translation hardware is updated from the page table 324 when references requiring address mappings not valid in the address translation hardware are encountered. In a particular embodiment, processor references to the page table 324 cause corresponding lines of the address translation hardware 320 to be invalidated as appropriate under a snoopy coherency protocol.

[0028] Within the TLB 320 (figure 1) I/O virtual addresses 244 (figure 2), which are generated by a DMA-capable peripheral such as intelligent peripherals 316 and 318 (figure 1), are received from an I/O bus 314 by an interface 242. These I/O virtual addresses 244 are separated into a group of page bits 204 and a group of offset bits 206. The page bits are looked up to find a matching address tag 200 in address mapping memory 201. Those fields

of address mapping memory 201 that are searched for matches, such as the address tag 200 and table location 220, may be implemented as content addressable memory (CAM) or may be implemented as a single or multiple-way set-associative memory system as known in the art. Each entry in the TLB 320 contains an entry-valid flag 202. Each intelligent peripheral 316 and 318 has an interface with DMA capability 317, and may include an embedded processor 319, and memory 321.

[0029] The TLB 320 is capable of storing several address mapping entries in an address mapping memory 201. Each address mapping entry comprises an address tag 200 (figure 2) field indicating which I/O virtual page the entry applies to, a memory page 222 field indicating the associated physical memory page in host memory, an entry valid flag 202, a table location 220 field containing the cacheline address at which the particular address mapping entry is stored in a page table in host memory, and other information.

[0030] References to addresses not found in the TLB, or references to addresses having an entry-valid flag 202 marked invalid, trigger an update of the TLB from a page table in memory 302. The update may be hardware-assisted or interrupt-driven as known in the art.

[0031] For addresses having an entry-valid flag 202 marked valid, the TLB provides a physical memory page field 222, from which high order address bits 208 are read that are concatenated with the offset bits 206 to form a complete physical memory address 210. The physical memory address is then transmitted to memory 102 or 302 over an interconnect 214 by a bus interface 212 of the bridge. Interconnect 214 may be a processor-memory (system) bus.

[0032] The TLB is fitted with coherency maintenance apparatus, which in the illustrated embodiment is snoop logic 216. Whenever system bus 214 performs a cacheline access, the cacheline address is checked against table location fields 220 of address mapping entries in the TLB. If a match is found between the bus address and the table location field 220 of any address mapping entry of the TLB, the corresponding entry valid flag is cleared if appropriate for the particular access being performed. The table location field may be implemented as a single or N-way set associative memory, or as a content addressable memory as known in the art.

[0033] The table location field 220 of each address mapping of a particular embodiment stores a cacheline address, such that any access that may modify the cacheline matches the address and causes invalidation of the associated TLB entry. In this

embodiment, the address tag 200, table location 220, and memory page 222 of a page table entry are all located in the same cacheline, therefore any potentially modifying access to any of these fields invalidates the associated TLB entry.

[0034] In an alternative embodiment, a computer system has at least one processor 100 (Figure 3) which communicates to a memory 102 through a system controller 106. System controller 106 incorporates coherency maintenance logic 108, in a particular embodiment the coherency protocol is directory-based. System controller 106 is also coupled to one, two, or more host channel adapters, such as host channel adapters (HCA) 110 and 112. Host channel adapters 110 and 112 each contain I/O address translation logic incorporating a TLB 114 and 116. Host channel adapters 110 and 112 also may each communicate with one or more Input-Output (I/O) devices, such as a keyboard and mouse interface 118, and one or more intelligent peripheral devices, such as intelligent peripheral 120 and Redundant Array of Independent Disks (RAID) or other disk controller 122. Each intelligent peripheral device has one or more embedded processors 124, memory 126, and peripheral specific controller apparatus 128. In the case of RAID controller 122, peripheral specific controller apparatus 128 may include SCSI controllers 130.

[0035] The intelligent peripherals 120 and 122 each couple through a Target Channel Adapter (TCA) 132. TCA 132 incorporates RDMA logic, such that each intelligent peripheral 120 can transfer blocks of data to or from memory 102 in DMA fashion.

[0036] In the illustrated embodiment, address translation logic 114 and 116 each contain a Translation Lookaside Buffer (TLB).

[0037] The TLB 114, 116 of the embodiment of Figure 3 is very similar to the TLB of Figure 2, except that snoop logic 216 is replaced by logic that interacts with a cache directory of the system; such that the TLB may acquire and release ownership of cachelines containing page table entries. It is anticipated that the cache directory also manages coherency of processor caches in the system.

[0038] It is anticipated that a system having host channel adapters with I/O address mapping hardware could be implemented with a snoop-based coherency maintenance apparatus, and that a system having a host bridge with I/O address mapping hardware could be implemented with a directory-based coherency maintenance apparatus.

[0039] While the invention has been particularly shown and described with reference to a preferred embodiment thereof, it will be understood by those skilled in the art that various other changes in the form and details may be made without departing from the spirit

and scope of the invention. It is to be understood that various changes may be made in adapting the invention to different embodiments without departing from the broader inventive concepts disclosed herein and comprehended by the claims that follow.